*The Journal of Navigation* (2021), **74**:2 343–359
doi:10.1017/S0373463320000582

CAMBRIDGE UNIVERSITY PRESS

**RESEARCH ARTICLE**

# A new visual/inertial integrated navigation algorithm based on sliding-window factor graph optimisation

Haiying Liu,[1]* Jingqi Wang,[2] Jianxin Feng,[1] and Xinyao Wang[2]

[1] College of Astronautics, Nanjing University of Aeronautics and Astronautics, Nanjing 210016, P. R. China
[2] College of Automation Engineering, Nanjing University of Aeronautics and Astronautics, Nanjing 210016, P. R. China.
*Corresponding author. E-mail: liuhaiying@nuaa.edu.cn

**Received:** 4 May 2020; **Accepted:** 12 October 2020; **First published online:** 1 December 2020

**Keywords:** visual–inertial navigation, InEKF, factor graph optimisation, sliding-window filter

**Abstract**

Visual–Inertial Navigation Systems (VINS) plays an important role in many navigation applications. In order to improve the performance of VINS, a new visual/inertial integrated navigation method, named Sliding-Window Factor Graph optimised algorithm with Dynamic prior information (DSWFG), is proposed. To bound computational complexity, the algorithm limits the scale of data operations through sliding windows, and constructs the states to be optimised in the window with factor graph; at the same time, the prior information for sliding windows is set dynamically to maintain interframe constraints and ensure the accuracy of the state estimation after optimisation. First, the dynamic model of vehicle and the observation equation of VINS are introduced. Next, as a contrast, an Invariant Extended Kalman Filter (InEKF) is constructed. Then, the DSWFG algorithm is described in detail. Finally, based on the test data, the comparison experiments of Extended Kalman Filter (EKF), InEKF and DSWFG algorithms in different motion scenes are presented. The results show that the new method can achieve superior accuracy and stability in almost all motion scenes.

## 1. Introduction

In many challenging environments, the satellite signal is weak or even blocked, which leads to the failure of satellite navigation systems. In order to solve this problem, a variety of other autonomous navigation methods can be used, including inertial navigation, visual navigation and so on. The inertial navigation option has the advantages of high output rate and not being affected by environmental interference, but its error will increase rapidly over time. The output rate of the visual sensor is low, but it can provide accurate information after image processing. A combination of visual and inertial can complement each other and improve autonomous navigation performance significantly in unknown and challenging environments. Therefore, because of their huge potential, Visual–Inertial Navigation Systems (VINS) have become a popular research option (Eckenhoff et al., 2017). According to the different back-end data-fusion processing methods, the VINS can be divided into two categories: one is the filtering-based method represented by the Kalman filter, which assumes a certain Markov property and the current state estimation only needs to consider the influence of the previous state; the other is the nonlinear optimisation algorithm represented by graph optimisation, which not only considers the state at the previous moment but also the states of a section or the whole (Huang, 2019).

The filtering-based VINS method generally uses the motion data collected by the inertial measurement unit (IMU) to predict the motion state. At the same time, this method processes the image data collected by the visual sensor, extracts the feature information as an observation and updates the state estimate. A tightly coupled visual–inertial navigation algorithm based on the extended Kalman filter (EKF) is

proposed (Mourikis and Roumeliotis, 2007), known as multi-state constraint kalman filter (MSCKF). This algorithm is one of the earliest successful VINS algorithms (Jiang et al., 2019). However, filtering-based methods have problems with accumulated linearisation errors and inconsistency (Hsiung et al., 2018). From the perspective of inconsistent estimates of VINS, a VINS method based on observability constraints is proposed (Hesch et al., 2014), and a real-time visual-inertial odometry algorithm based on EKF is proposed (Li and Mourikis, 2013) which uses first-estimate Jacobian (FEJ) method (Huang et al., 2009) to improve the MSCKF algorithm by enforcing fixed linearisation points.

With the improvement of computer performance, the VINS methods based on nonlinear optimisation can linearly estimate the motion states multiple times and theoretically obtain the optimal estimate of all the states to be optimised. Due to the above advantages, the VINS methods based on nonlinear optimisation have attracted the attention of researchers. Nevertheless, the large amount of data accumulation will affect the real-time performance of the system. Regarding the calculation scale problem brought by the nonlinear optimisation method, some scholars have used the sliding-window filter (SWF) to improve the real-time performance of the system by restricting the amount of data through the use of sliding windows (Sibley et al., 2006; Zhuang et al., 2019). But a big problem with SWF is how to deal with the discarded state. It is proposed that abandoning the measurement quantity outside the sliding window would lead to loss of information related to the interaction variables (Sibley et al., 2010), and a reasonable way to discard the old state from the sliding window is to marginalise the discarded state (Dong-Si and Mourikis, 2011). This would marginalise the prior discarded frame in the sliding window and estimate the Gaussian distribution of the states in the sliding window at this time by adding the prior information.

At the same time, it should be noted that taking the marginalisation operation blindly will introduce additional prior information and destroy the sparse nature of the matrix (Kretzschmar et al., 2011; Johannsson et al., 2013; Hsiung et al., 2018; Wilbers et al., 2019). A method is proposed to avoid introducing new constraints between existing nodes, so that the number of variables only grows with the size of the exploration space (Johannsson et al., 2013). An information-based criterion for determining which nodes will be marginalised in pose-graph optimisation is proposed (Kretzschmar et al., 2011). To avoid introducing additional prior information, it is necessary to ensure reduction of the constraints of the nodes to be marginalised. Therefore, it is necessary to design complicated algorithms to discriminate suitable nodes for the marginalisation operation. A sparse scheme is designed, which focuses on reducing the constraints brought by marginalisation and optimally estimating the location of landmarks without affecting the sparse pattern of the problem (Wilbers et al., 2019). In addition, a nonlinear factor graph is used to sparse and edge the dense prior information, and the resulting factor graph maintained information sparsity (Hsiung et al., 2018).

This paper proposes a novel visual–inertial integrated navigation method by building prior information of sliding window dynamically, named SWF graph optimised algorithm with dynamic prior information (DSWFG). The DSWFG algorithm uses sliding window to limit the data operation scale and constructs the fixed number of states in the window with the factor graph. At the same time, the prior information of the sliding window is set dynamically to maintain the inter-frame constraint. In this way, DSWFG algorithm can dynamically construct the prior information of the sliding window and constrain the states to be optimised in the window. Compared with using the marginalisation method and making the constructed graph densely connected, this method can avoid making the sparse matrix dense.

In this paper, the comparative experiment of optimal filtering and factor graph optimisation is designed to verify the effect of the proposed algorithm. Before introducing the DSWFG algorithm in detail, the invariant extended Kalman filter (InEKF) algorithm based on the Lie group is introduced. In the experimental part, comparative experiments are carried out in different motion scenarios. The results are analysed from the aspects of motion trajectory estimation, pose error statistics, and single-cycle time consumption. This contribution is organised as follows: Section 2 describes the space state model of mobile robots. Sections 3 and 4 describes the InEKF algorithm and the DSWFG algorithm, respectively. Section 5 presents a comparative analysis of EKF, InEKF and our new method in different motion scenarios. Finally, Section 6 contains the summary and conclusions of this contribution.

https://doi.org/10.1017/S0373463320000582 Published online by Cambridge University Press

## 2. Spatial state model of mobile vehicle

Assuming that there are a certain number of fixed landmarks in the real coordinate system, an mobile vehicle equipped with a monocular camera and an IMU tracks the fixed landmarks, $p$. Therefore, the estimated status including the rotation matrix $\boldsymbol{R}$, velocity $\boldsymbol{v}$, displacement $\boldsymbol{p}$, IMU deviation $\boldsymbol{b}_n = [\boldsymbol{w}_{n,b}, \boldsymbol{a}_{n,b}]$ and three-dimensional position $\boldsymbol{l}_i = [\boldsymbol{l}_1, \ldots, \boldsymbol{l}_p]$of the landmarks, then the dynamic model of the system can be constructed as

$$\begin{cases} \boldsymbol{R}_n = \boldsymbol{R}_{n-1} \exp((\boldsymbol{w}_{n,\mathrm{i}} - \boldsymbol{w}_{n,b} + \boldsymbol{n}_n^{\mathrm{w}})\mathrm{d}t) \\ \boldsymbol{v}_n = \boldsymbol{v}_{n-1} + (\boldsymbol{R}_n(\boldsymbol{a}_{n,\mathrm{i}} - \boldsymbol{a}_{n,b} + \boldsymbol{n}_n^{\mathrm{a}}) + {}^{\mathrm{W}}\boldsymbol{g})\mathrm{d}t \\ \boldsymbol{p}_n = \boldsymbol{p}_{n-1} + \boldsymbol{v}_n \mathrm{d}t \\ \boldsymbol{w}_{n,\mathrm{b}} = \boldsymbol{w}_{n-1,b} + \boldsymbol{n}_n^{\mathrm{bw}} \\ \boldsymbol{a}_{n,\mathrm{b}} = \boldsymbol{a}_{n-1,b} + \boldsymbol{n}_n^{\mathrm{ba}} \end{cases} \tag{2.1}$$

where $n$ = discrete time index; $\boldsymbol{R}_n$ = rotation matrix at epoch $n$; $\boldsymbol{v}_n$ = speed of the vehicle at epoch $n$; $\boldsymbol{p}_n$ = displacement of the agent at epoch $n$; $\exp(\boldsymbol{w}dt)$ maps the angular velocity $\boldsymbol{w}$ to the rotation matrix; $\boldsymbol{w}_{n,\mathrm{i}}$ and $\boldsymbol{a}_{n,\mathrm{i}}$ = angular velocity and acceleration output by the IMU at epoch $n$; ${}^{\mathrm{W}}\boldsymbol{g} = [\,0\ 0\ -\mathrm{g}\,]^{\mathrm{T}}$ = gravity, and $g$ = local acceleration of gravity; and $\mathrm{d}t$ = time difference between the measured moments.

The system noise model of Gaussian white noise is expressed as

$$\boldsymbol{n} = [\boldsymbol{n}_n^{\mathrm{w}}, \boldsymbol{n}_n^{\mathrm{a}}, \boldsymbol{n}_n^{\mathrm{bw}}, \boldsymbol{n}_n^{\mathrm{ba}}] \sim \mathrm{N}(0, \boldsymbol{Q}) \tag{2.2}$$

Constitute a special Euclidean group matrix $\boldsymbol{X}_n \in SE_{2+m}(3)$ with $\boldsymbol{R}$, $\boldsymbol{v}$, $\boldsymbol{p}$, $\boldsymbol{l}_i$, as follows

$$\boldsymbol{X}_n = \begin{bmatrix} \boldsymbol{R}_n & \boldsymbol{v}_n & \boldsymbol{p}_n & \boldsymbol{l}_i \\ \boldsymbol{0}_{2+m,3} & & \boldsymbol{I}_{2+m,2+m} & \end{bmatrix} \tag{2.3}$$

Due to the addition not being closed in manifold space $\boldsymbol{X}_n$, it must use exponential mapping for incremental description. However, the vector $\boldsymbol{b}_n$ is the additional IMU deviation satisfying the European addition. Therefore, the system state is expressed by $[\boldsymbol{X}_n, \boldsymbol{b}_n]$, and the system state equation is expressed as

$$[\boldsymbol{X}_n, \boldsymbol{b}_n] = f(\boldsymbol{X}_{n-1}, \boldsymbol{U}_{n-1} - \boldsymbol{b}_{n-1}, \boldsymbol{n}) \tag{2.4}$$

where $\boldsymbol{U}_n = [\boldsymbol{w}_{n,\mathrm{i}}, \boldsymbol{a}_{n,\mathrm{i}}]$ = input of the time-varying system; $\boldsymbol{n}$ = system noise.

The observation values of landmarks are $\boldsymbol{Z}_n^i (i = 1, \ldots, p)$ and the observation noise are $\boldsymbol{n}_{\mathrm{y}}^i$ $(i = 1, \ldots, p)$. The observation equation is expressed as

$$\boldsymbol{Z}_n^i = Y(\boldsymbol{X}_n) = \prod \left[ (\boldsymbol{R}_{\mathrm{c}}(\boldsymbol{R}_n^{\mathrm{T}}(\boldsymbol{l}_i - \boldsymbol{p}_n) - \boldsymbol{P}_{\mathrm{c}})) + \boldsymbol{n}_{\mathrm{y}}^i \right. \tag{2.5}$$

where $\prod$ = internal parameter matrix of the pinhole camera, $\boldsymbol{R}_{\mathrm{c}}$ and $\boldsymbol{P}_{\mathrm{c}}$ = rotation matrix and the translation vector from the IMU coordinate system to the camera coordinate system, respectively. The $\boldsymbol{n}_{\mathrm{y}} \sim N(0, \boldsymbol{N})$ is observation noise. For convenience of description, the Equation (2.5) is abbreviated as

$$\boldsymbol{Z}_n = \gamma(\boldsymbol{R}_n^{\mathrm{T}}(\boldsymbol{l} - \boldsymbol{p}_n)) + \boldsymbol{n}_{\mathrm{y}} \tag{2.6}$$

## 3. The optimal filtering algorithm

Based on the space state model of the robot, this section introduces the InEKF algorithm based on the filtering method. The traditional EKF algorithm estimates the system state by linearising the dynamic equations, but in the process of error transmission, the error transfer matrix $\boldsymbol{F}$ depends on the estimated value of the current state. When noise is introduced, the state estimation value could be unpredictable.

https://doi.org/10.1017/S0373463320000582 Published online by Cambridge University Press

Therefore, it may lead to filter divergence (Barrau, 2015). The InEKF algorithm changes the definition of state error by introducing the concept of the Lie group and Lie algebra, and reconstructs the error state with a special Euclidean group. Using the reconstructed error state to derive the error transfer equation in the Lie group space, the error transfer matrix $\boldsymbol{F}$ is relatively independent of the state estimate (Barrau, 2015). To a certain extent, the InEKF algorithm can solve the accumulated linearisation errors and inconsistencies (Wu et al., 2017).

### *3.1. Status prediction*

Based on the error transmission, the system equation of the error $\boldsymbol{e}_n$ can be derived and the state error can be defined as

$$\boldsymbol{e}_n = (\hat{\boldsymbol{X}}_n \boldsymbol{X}_n^{-1}, \hat{\boldsymbol{b}}_n - \boldsymbol{b}_n) \equiv (\boldsymbol{\eta}_n, \boldsymbol{\varsigma}_n) \tag{3.1}$$

where the symbol $\wedge$ = the observed value of the state $\boldsymbol{X}_n$; and the symbol $\equiv$ indicates that the error $\boldsymbol{e}_n$ can be defined as the form of $\boldsymbol{\eta}_n$ and $\boldsymbol{\varsigma}_n$, as

$$\boldsymbol{\varsigma}_n = \begin{bmatrix} \hat{\boldsymbol{w}}_{n,b} - \boldsymbol{w}_{n,b} \\ \hat{\boldsymbol{a}}_{n,b} - \boldsymbol{a}_{n,b} \end{bmatrix} \equiv \begin{bmatrix} \boldsymbol{\varsigma}_n^{\mathrm{w}} \\ \boldsymbol{\varsigma}_n^{\mathrm{a}} \end{bmatrix},$$

$$\boldsymbol{\eta}_n = \begin{bmatrix} \hat{\boldsymbol{R}}_n \boldsymbol{R}_n^{\mathrm{T}} & \hat{\boldsymbol{v}}_n - \hat{\boldsymbol{R}}_n \boldsymbol{R}_n^{\mathrm{T}} \boldsymbol{v}_n & \hat{\boldsymbol{p}}_n - \hat{\boldsymbol{R}}_n \boldsymbol{R}_n^{\mathrm{T}} \boldsymbol{p}_n & \hat{\boldsymbol{l}}_i - \hat{\boldsymbol{R}}_n \boldsymbol{R}_n^{\mathrm{T}} \boldsymbol{l}_i \\ \boldsymbol{0}_{2+p,3} & & \boldsymbol{I}_{2+m,2+m} & \end{bmatrix} = \begin{bmatrix} \boldsymbol{\eta}_{R_n} & \boldsymbol{\xi}_{v_n} & \boldsymbol{\xi}_{p_n} & \boldsymbol{\xi}_{l_n} \\ \boldsymbol{0}_{2+m,3} & & \boldsymbol{I}_{2+m,2+m} & \end{bmatrix}$$

Let $\boldsymbol{\eta}_{R_n} = \hat{\boldsymbol{R}}_n \boldsymbol{R}_n^{\mathrm{T}}$, and the lie algebra of $\boldsymbol{\eta}_{R_n} = \boldsymbol{\xi}_{R_n}$, then $\boldsymbol{\eta}_{R_n} = \exp(\boldsymbol{\xi}_{R_n})$; Let $\boldsymbol{\xi}_{v_n} = \hat{\boldsymbol{v}}_n - \hat{\boldsymbol{R}}_n \boldsymbol{R}_n^{\mathrm{T}} \boldsymbol{v}_n$, $\boldsymbol{\xi}_{p_n} = \hat{\boldsymbol{p}}_n - \hat{\boldsymbol{R}}_n \boldsymbol{R}_n^{\mathrm{T}} \boldsymbol{p}_n$, $\boldsymbol{\xi}_{l_n} = \hat{\boldsymbol{l}}_i - \hat{\boldsymbol{R}}_n \boldsymbol{R}_n^{\mathrm{T}} \boldsymbol{l}_i$. Since $\boldsymbol{\eta}_{R_n}$ is a small quantity, the exponential function can be approximated to $\boldsymbol{\eta}_{R_n} = \exp(\boldsymbol{\xi}_{R_n}) \approx 1 + (\boldsymbol{\xi}_{R_n})_\times$ by first-order Taylor approximation, then

$$\dot{\boldsymbol{\eta}}_n = \begin{bmatrix} (\dot{\boldsymbol{\xi}}_{R_n})_\times & \dot{\boldsymbol{\xi}}_{v_n} & \dot{\boldsymbol{\xi}}_{p_n} & \dot{\boldsymbol{\xi}}_{l_n} \\ & \boldsymbol{0}_{2+m,5+m} & & \end{bmatrix} \tag{3.2}$$

where $(\cdot)_\times$ is expressed as a skew symmetric matrix.

$$\begin{cases} (\dot{\boldsymbol{\xi}}_{R_n})_\times = (\hat{\boldsymbol{R}}_n(\boldsymbol{n}_n^{\mathrm{w}} - \boldsymbol{\varsigma}_n^{\mathrm{w}}))_\times \\ \dot{\boldsymbol{\xi}}_{v_n} = (g)_\times \boldsymbol{\xi}_{R_n}^{\mathrm{T}} + \hat{\boldsymbol{R}}_n(\boldsymbol{n}_n^{\mathrm{w}} - \boldsymbol{\varsigma}_n^{\mathrm{a}}) + (\hat{\boldsymbol{v}}_n)_\times \hat{\boldsymbol{R}}_n(\boldsymbol{n}_n^{\mathrm{w}} - \boldsymbol{\varsigma}_n^{\mathrm{w}}) \\ \dot{\boldsymbol{\xi}}_{p_n} = \boldsymbol{\xi}_{v_n} + (\hat{\boldsymbol{p}}_n)_\times \hat{\boldsymbol{R}}_n(\boldsymbol{n}_n^{\mathrm{w}} - \boldsymbol{\varsigma}_n^{\mathrm{w}}) \\ \dot{\boldsymbol{\xi}}_{l_n} = (\hat{\boldsymbol{l}}_i)_\times \hat{\boldsymbol{R}}_n(\boldsymbol{n}_n^{\mathrm{w}} - \boldsymbol{\varsigma}_n^{\mathrm{w}}) \\ \dot{\boldsymbol{\varsigma}}_n^{\mathrm{w}} = -\boldsymbol{n}_n^{\mathrm{bw}} \\ \dot{\boldsymbol{\varsigma}}_n^{\mathrm{a}} = -\boldsymbol{n}_n^{\mathrm{ba}} \end{cases} \tag{3.3}$$

where $\boldsymbol{n}_n^{\mathrm{w}}$ = noise measured by the gyroscope, and $\boldsymbol{n}_n^{\mathrm{a}}$ = noise measured by the accelerometer. Let $\boldsymbol{\xi}_n = [\boldsymbol{\xi}_{R_n}^{\mathrm{T}}, \boldsymbol{\xi}_{v_n}^{\mathrm{T}}, \boldsymbol{\xi}_{p_n}^{\mathrm{T}}, \boldsymbol{\xi}_{l_n}^{\mathrm{T}}]^{\mathrm{T}}$, $\boldsymbol{n}_n = [\boldsymbol{n}_n^{\mathrm{w}}, \boldsymbol{n}_n^{\mathrm{a}}, \boldsymbol{n}_n^{\mathrm{bw}}, \boldsymbol{n}_n^{\mathrm{ba}}]^{\mathrm{T}}$, then the error state transfer equation is given as

$$\frac{d}{dn}\left(\begin{bmatrix} \boldsymbol{\xi}_n \\ \boldsymbol{\varsigma}_n \end{bmatrix}\right) = \boldsymbol{F}_n \begin{bmatrix} \boldsymbol{\xi}_n \\ \boldsymbol{\varsigma}_n \end{bmatrix} + \boldsymbol{G}_n \boldsymbol{n}_n \tag{3.4}$$

Derive the state transition matrix $\boldsymbol{Fn}$ and calculate the covariance matrix $\boldsymbol{Pn}$, that is

$$\boldsymbol{\Phi}_n = \exp(\boldsymbol{F}_n \mathrm{d}t) \tag{3.5}$$

$$\boldsymbol{P}_n = \boldsymbol{\Phi}_n \boldsymbol{P}_{n-1} \boldsymbol{\Phi}_n^{\mathrm{T}} + \boldsymbol{G}_n \boldsymbol{Q}_n \boldsymbol{G}_n^{\mathrm{T}} \tag{3.6}$$

https://doi.org/10.1017/S0373463320000582 Published online by Cambridge University Press

### *3.2. Measurement update*

According to the observation Equation (2.6), there is

$$\begin{cases} \boldsymbol{y}_n^1 = \boldsymbol{R}_n^{\mathrm{T}}(\boldsymbol{l}_1 - \boldsymbol{p}_n)) + \boldsymbol{n}_{\mathrm{y}}^1 \\ \boldsymbol{y}_n^2 = \boldsymbol{R}_n^{\mathrm{T}}(\boldsymbol{l}_2 - \boldsymbol{p}_n)) + \boldsymbol{n}_{\mathrm{y}}^2 \\ \cdots\cdots \\ \boldsymbol{y}_n^k = \boldsymbol{R}_n^{\mathrm{T}}(\boldsymbol{l}_k - \boldsymbol{p}_n)) + \boldsymbol{n}_{\mathrm{y}}^k \end{cases} \tag{3.7}$$

where $\boldsymbol{y}_n^i$ = distance of the landmark $i$ relative to the sensor itself. And one line of the Equation (3.7) can be rewrite as

$$\boldsymbol{Y}_n^i = \boldsymbol{X}_n^{-1}\boldsymbol{L}^i + \boldsymbol{S}_n^i \tag{3.8}$$

where $\boldsymbol{Y}_n^i = (\boldsymbol{y}_n^i, 0, 1, 0)^{\mathrm{T}}$, $\boldsymbol{L}^i = (\boldsymbol{l}_n^i, 0, 1, 0)^{\mathrm{T}}$, $\boldsymbol{S}_n^i = (\boldsymbol{n}_{\mathrm{y}}^i, 0, 0, 0)^{\mathrm{T}}$.

Then the observation error $\tilde{\boldsymbol{Z}}_n$ is defined as

$$\begin{aligned} \boldsymbol{Z}_n^i &= \hat{\boldsymbol{X}}_n\boldsymbol{Y}_n^i - \boldsymbol{L}^i \\ &= \boldsymbol{\eta}_n\boldsymbol{L}^i - \boldsymbol{L}^i + \hat{\boldsymbol{X}}_n\boldsymbol{S}_n^i \end{aligned} \tag{3.9}$$

Stack up all the observation errors and eliminate all zero rows, there is

$$\begin{aligned} \tilde{\boldsymbol{Z}}_n &= \begin{pmatrix} (\boldsymbol{\xi}_{R_n})_\times \boldsymbol{l}_n^1 + \boldsymbol{\xi}_{p_n} + \hat{\boldsymbol{R}}_n\boldsymbol{n}_{\mathrm{y}}^1 \\ \cdots\cdots \\ (\boldsymbol{\xi}_{R_n})_\times \boldsymbol{l}_n^k + \boldsymbol{\xi}_{p_n} + \hat{\boldsymbol{R}}_n\boldsymbol{n}_{\mathrm{y}}^k \end{pmatrix} \\ &\approx \begin{pmatrix} -\left(\boldsymbol{l}_n^1\right)_\times \boldsymbol{L}^1 + \hat{\boldsymbol{X}}_n\boldsymbol{S}_n^1 \\ \cdots\cdots \\ -\left(\boldsymbol{l}_n^k\right)_\times \boldsymbol{L}^k + \hat{\boldsymbol{X}}_n\boldsymbol{S}_n^k \end{pmatrix} \begin{pmatrix} \boldsymbol{\xi}_{R_n} \\ \boldsymbol{\xi}_{v_n} \\ \boldsymbol{\xi}_{p_n} \\ \boldsymbol{\xi}_{l_n} \end{pmatrix} + \begin{pmatrix} \hat{\boldsymbol{R}}_n\boldsymbol{n}_{\mathrm{y}}^1 \\ \cdots\cdots \\ \hat{\boldsymbol{R}}_n\boldsymbol{n}_{\mathrm{y}}^k \end{pmatrix} \\ &= \boldsymbol{H}_n\boldsymbol{\xi}_n + \boldsymbol{n}_{\mathrm{y}} \end{aligned} \tag{3.10}$$

and the error after update is

$$\boldsymbol{\eta}_{n+1} = \hat{\boldsymbol{X}}_{n+1}\boldsymbol{X}_n^{-1} = \hat{\boldsymbol{X}}_{n+1}\hat{\boldsymbol{X}}_n^{-1}\boldsymbol{\eta}_n \tag{3.11}$$

Then the error state update equation can be established as

$$\boldsymbol{\eta}_{n+1} = \exp(\boldsymbol{K}^{\xi}\,\tilde{\boldsymbol{Z}}_n)\boldsymbol{\eta}_n \tag{3.12}$$

$\boldsymbol{\eta}_n = \exp(\boldsymbol{\xi}_n) \approx 1 + (\boldsymbol{\xi}_n)_\times$, ignore the high-order small quantity, there is

$$\boldsymbol{\xi}_{n+1} = \boldsymbol{\xi}_n + \boldsymbol{K}^{\xi}\tilde{\boldsymbol{Z}}_n \tag{3.13}$$

Therefore, the state error is updated as

$$\begin{bmatrix} \boldsymbol{\xi}_{n+1} \\ \boldsymbol{\varsigma}_{n+1} \end{bmatrix} = \begin{bmatrix} \boldsymbol{\xi}_n \\ \boldsymbol{\varsigma}_n \end{bmatrix} + \boldsymbol{K}\left(\begin{bmatrix} \boldsymbol{H}_{\xi_n} & \boldsymbol{H}_{\varsigma_n} \end{bmatrix}\begin{bmatrix} \boldsymbol{\xi}_n \\ \boldsymbol{\varsigma}_n \end{bmatrix} + \boldsymbol{V}_n\right) \tag{3.14}$$

where $\boldsymbol{K} = [\boldsymbol{K}^{\xi}, \boldsymbol{K}^{\varsigma}]^{\mathrm{T}}$; $\boldsymbol{H}_{\xi_n} = \boldsymbol{H}_n$; $\boldsymbol{H}_{\varsigma_n} = \begin{bmatrix} \boldsymbol{0}_{3k,3} & \boldsymbol{0}_{3k,3} \end{bmatrix}$; $\boldsymbol{V}_n$ is the Gaussian noise.

https://doi.org/10.1017/S0373463320000582 Published online by Cambridge University Press

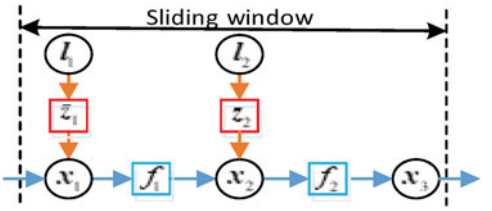


***Figure 1.*** *Express the VINS movement process using factor graphs.*

### *3.3. Filter update*

The gain $\boldsymbol{K}$ can be calculated as

$$\boldsymbol{K} = \boldsymbol{P}_n \boldsymbol{H}_n^{\mathrm{T}} (\boldsymbol{H}_n \boldsymbol{P}_n \boldsymbol{H}_n^{\mathrm{T}} + \boldsymbol{V}_n)^{-1} \tag{3.15}$$

and the state deviation increment is calculated through the gain $\boldsymbol{K}$

$$[\delta \boldsymbol{X}, \delta \boldsymbol{b}] = \boldsymbol{K} \tilde{\boldsymbol{Z}}_n \tag{3.16}$$

Then the system states $\boldsymbol{X}_{n+1}$ and covariance $\boldsymbol{P}_{n+1}$ are updated as

$$\begin{cases} \boldsymbol{X}_{n+1} = \exp(\delta \boldsymbol{X}) \boldsymbol{X}_n \\ \boldsymbol{b}_{n+1} = \boldsymbol{b}_n + \delta \boldsymbol{b} \\ \boldsymbol{P}_{n+1} = (\boldsymbol{I} - \boldsymbol{K} \boldsymbol{H}_n) \boldsymbol{P}_n \end{cases} \tag{3.17}$$

## 4. The DSWFG algorithm

### *4.1. The overall framework*

Based on the state Equation (2.4) and observation Equation (2.5), a small segment of the VINS movement process is established with a factor graph, as shown in Figure 1. In this figure, the pose $\boldsymbol{x}_n$ of the system and the space coordinates $\boldsymbol{l}_n$ of the landmarks are variable nodes, represented by circles; the state constraints $\boldsymbol{f}_n$ and observation constraints $z_n$ of adjacent variable nodes are factor nodes, represented by squares. Because IMU has a high data rate, it is generally regarded as a reference source in the VINS. When the visual or IMU sensors update the observation data, the variable nodes (for example, $x_1$) representing the state variables at the current time and the constrained factor nodes (for example, $z_1$) are generated in the factor graph. During the operation of the factor graph optimisation, whenever new observations are obtained, the Bayesian inference can be used to obtain the optimal estimate of the variable at each time.

Therefore, the complete motion process of VINS can be represented by a factor graph as several dots and several edges connecting the dots. Each dot in the graph is the variable node to be optimised, and each edge connected to two variable nodes corresponds to a factor node, which is used to constrain the variable node. Through the construction of dots and edges, the error distribution structure of the visual–inertial system can be established intuitively, and the constraint relationship between variables can be displayed intuitively. For the factor graphs, its maximum posterior probability inference is equivalent to maximising the product of the potential energy of all factors (Dellaert and Kaess, 2017), which is also equivalent to minimising the solution of the nonlinear least squares problem. For the established graph, the numerical iterative method, like Gauss–Newton (GN) or Levenberg–Marquardt (LM), can be used to obtain the optimal solution of each factor node through a series of linear approximations.

Based on the factor graph optimisation, an improved VINS method named DSWFG is proposed. The following elaborates the DSWFG from three parts: the overall framework, the front-end processing method and the back-end processing method. Whenever there is an image frame input, the front-end part

https://doi.org/10.1017/S0373463320000582 Published online by Cambridge University Press

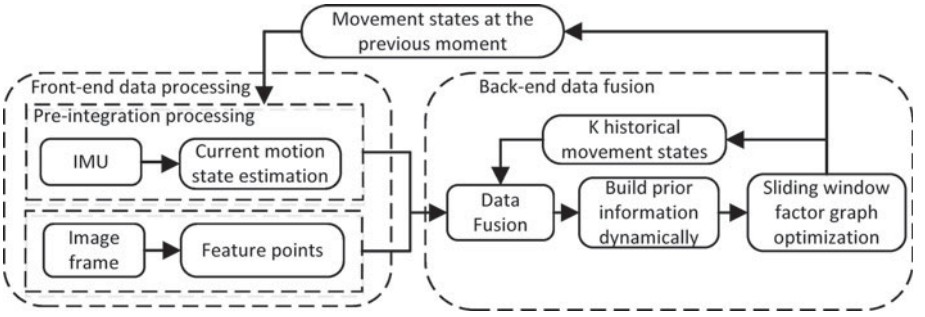


***Figure 2.*** *The overall framework of DSWFG algorithm.*

extracts the speeded up robust features (SURF) points on the image frame and uses the kanade-lucas-tomasi (KLT) tracking algorithm to track the feature points extracted from the previous image frame and takes the threshold constraint method to obtain the valid feature points of the current frame. At the same time, the front-end part uses the IMU pre-integration method to align the IMU with the image frame, then sends the processed IMU data and vision data to the back end. The back-end part limits the data operation scale with SWF and constructs the graph with the fixed number of states in the window using factor graphs. At the same time, the prior information of the sliding window is set dynamically to maintain the interframe constraint. The overall framework of DSWFG algorithm is shown in Figure 2. The following will be illustrated from the IMU pre-integration method in the front-end part, the construction of prior information of the sliding window in the back-end part and the optimisation of the states in the window.

### *4.2. The IMU pre-integration method in the front-end part*

Due to the high frequency of the IMU, there will be a large number of variable nodes added in the graph. Using the numerical iterative method to approximate the optimal solution of each factor node directly will result in less optimisation benefits for the high computational complexity. Therefore, a set of the IMU measurements between two adjacent image frames are integrated into a single relative motion constraint by the IMU pre-integration, and the processed IMU measurements are aligned with the image frames. Then, in the process of overall batch optimisation, the visual and IMU constraints can be considered at the same time, reducing the calculation burden and improving the optimisation effect. In addition, the IMU pre-integration method introduces the pre-integration term through an incremental expression, which can effectively avoid the recalculation of the IMU measurements during the optimisation process. The IMU pre-integration method in this section is based on the research of Forster (Forster et al., 2017), and the derivation process is as follows.

The motion relationship and the IMU noise model can be established as

$$\begin{cases} \tilde{\boldsymbol{w}}_{\mathrm{B}}(t) = \boldsymbol{w}_{\mathrm{B}}(t) + \boldsymbol{b}_g(t) + \boldsymbol{\eta}_g(t) \\ \tilde{\boldsymbol{a}}_{\mathrm{B}}(t) = \boldsymbol{R}_{\mathrm{WB}}^{\mathrm{T}}(\boldsymbol{a}_{\mathrm{W}}(t) - \boldsymbol{g}_{\mathrm{W}}) + \boldsymbol{b}_a(t) + \boldsymbol{\eta}_a(t) \end{cases} \tag{4.1}$$

where $\boldsymbol{R}_{\mathrm{WB}}$ represents rotation matrix from the carrier coordinate system B to world coordinate system W. IMU angular velocity measurements $\boldsymbol{w}_{\mathrm{B}}$ are added bias $\boldsymbol{b}_g$ and noise $\boldsymbol{\eta}_g$. The measured values of IMU accelerometer $\tilde{\boldsymbol{a}}_{\mathrm{B}}$ are added bias $\boldsymbol{b}_a$ and noise $\boldsymbol{\eta}_a$. The differential motion model is established as

$$\begin{cases} \dot{\boldsymbol{R}}_{\mathrm{WB}} = \boldsymbol{R}_{\mathrm{WB}} \boldsymbol{w}_{\mathrm{B}}^{\wedge} \\ \dot{\boldsymbol{v}}_{\mathrm{W}} = \boldsymbol{a}_{\mathrm{W}} \\ \dot{\boldsymbol{p}}_{\mathrm{W}} = \boldsymbol{v}_{\mathrm{W}} \end{cases} \tag{4.2}$$

https://doi.org/10.1017/S0373463320000582 Published online by Cambridge University Press

where the symbol $\wedge$ represents the transformation of the vector into an antisymmetric matrix. The discrete form of motion equation is obtained by Euler integral

$$\begin{cases} \boldsymbol{R}_{\mathrm{WB}}(t+\Delta t) = \boldsymbol{R}_{\mathrm{WB}}(t)\mathrm{Exp}(\boldsymbol{w}_{\mathrm{B}}(t)\Delta t) \\ \boldsymbol{v}_{\mathrm{W}}(t+\Delta t) = \boldsymbol{v}_{\mathrm{W}}(t) + \boldsymbol{a}_{\mathrm{W}}(t)\Delta t \\ \boldsymbol{p}_{\mathrm{W}}(t+\Delta t) = \boldsymbol{p}_{\mathrm{W}}(t) + \boldsymbol{v}_{\mathrm{W}}(t)\Delta t + \dfrac{1}{2}\boldsymbol{a}_{\mathrm{W}}(t)\Delta t^2 \end{cases} \tag{4.3}$$

where $\Delta t$ = interval time between two adjacent IMU measurements. Combining the IMU noise model with Equation (4.3) gives

$$\begin{cases} \boldsymbol{R}_{\mathrm{WB}}(t+\Delta t) = \boldsymbol{R}_{\mathrm{WB}}(t)\mathrm{Exp}\left(\left(\tilde{\boldsymbol{w}}_{\mathrm{B}}(t) - \boldsymbol{b}_g(t) - \boldsymbol{\eta}_{gd}(t)\right)\Delta t\right) \\ \boldsymbol{v}_{\mathrm{W}}(t+\Delta t) = \boldsymbol{v}_{\mathrm{W}}(t) + \boldsymbol{g}_{\mathrm{W}}\Delta t + \boldsymbol{R}_{\mathrm{WB}}\left(\tilde{\boldsymbol{a}}_{\mathrm{B}}(t) - \boldsymbol{b}_a(t) - \boldsymbol{\eta}_{ad}(t)\right)\Delta t \\ \boldsymbol{p}_{\mathrm{W}}(t+\Delta t) = \boldsymbol{p}_{\mathrm{W}}(t) + \boldsymbol{v}_{\mathrm{W}}(t)\Delta t + \dfrac{1}{2}\boldsymbol{g}_{\mathrm{W}}\Delta t^2 + \dfrac{1}{2}\boldsymbol{R}_{\mathrm{WB}}\left(\tilde{\boldsymbol{a}}_{\mathrm{B}}(t) - \boldsymbol{b}_a(t) - \boldsymbol{\eta}_{ad}(t)\right)\Delta t^2 \end{cases} \tag{4.4}$$

where noise terms $\boldsymbol{\eta}_{gd}$ and $\boldsymbol{\eta}_{ad}$ are in discrete form, and the relationships between discrete noise and continuous noise are

$$\begin{cases} \mathrm{Cov}(\boldsymbol{\eta}_{gd}(t)) = \dfrac{1}{\Delta t}\mathrm{Cov}(\boldsymbol{\eta}_g(t)) \\ \mathrm{Cov}(\boldsymbol{\eta}_{ad}(t)) = \dfrac{1}{\Delta t}\mathrm{Cov}(\boldsymbol{\eta}_a(t)) \end{cases} \tag{4.5}$$

For symbol simplicity, the reference coordinate system in the formula is omitted. Assuming that the IMU measurements are aligned with the image frames, then the pose relationship between the adjacent image frames $i$ and $j$ can be obtained as

$$\begin{cases} \boldsymbol{R}_j = \boldsymbol{R}_j \prod_{k=i}^{j-1} \mathrm{Exp}\left(\left(\tilde{\boldsymbol{w}}_k - \boldsymbol{b}_k^g - \boldsymbol{\eta}_k^{gd}\right)\Delta t\right) \\ \boldsymbol{v}_j = \boldsymbol{v}_i + \boldsymbol{g}\Delta t_{ij} + \sum_{k=i}^{j-1} \boldsymbol{R}_k\left(\tilde{\boldsymbol{a}}_k - \boldsymbol{b}_k^a - \boldsymbol{\eta}_k^{ad}\right)\Delta t \\ \boldsymbol{p}_j = \boldsymbol{p}_i + \sum_{k=i}^{j-1}\left[\boldsymbol{v}_k\Delta t + \dfrac{1}{2}\boldsymbol{g}\Delta t^2 + + \dfrac{1}{2}\boldsymbol{R}_k\left(\tilde{\boldsymbol{a}}_k - \boldsymbol{b}_k^a - \boldsymbol{\eta}_k^{ad}\right)\Delta t^2\right] \end{cases} \tag{4.6}$$

In order to avoid solving $\boldsymbol{R}_i$, $\boldsymbol{v}_i$, $\boldsymbol{p}_i$ again, introduce the pre-integration term through incremental expression

$$\begin{cases} \Delta\boldsymbol{R}_{ij} = \boldsymbol{R}_i^{\mathrm{T}}\boldsymbol{R}_i = \prod_{k=i}^{j-1} \mathrm{Exp}\left(\left(\tilde{\boldsymbol{w}}_k - \boldsymbol{b}_k^g - \boldsymbol{\eta}_k^{gd}\right)\Delta t\right) \\ \Delta\boldsymbol{v}_{ij} = \boldsymbol{R}_i^{\mathrm{T}}(\boldsymbol{v}_j - \boldsymbol{v}_i - \boldsymbol{g}\Delta t_{ij}) = \sum_{k=i}^{j-1} \boldsymbol{R}_{ik}\left(\tilde{\boldsymbol{a}}_k - \boldsymbol{b}_k^a - \boldsymbol{\eta}_k^{ad}\right)\Delta t \\ \Delta\boldsymbol{p}_{ij} = \boldsymbol{R}_i^{\mathrm{T}}\left(\boldsymbol{p}_j - \boldsymbol{p}_i - \boldsymbol{v}_i\Delta t_{ij} + \dfrac{1}{2}\boldsymbol{g}\Delta t_{ij}^2\right) = \sum_{k=i}^{j-1}\left[\boldsymbol{v}_{ik}\Delta t + \dfrac{1}{2}\boldsymbol{R}_{ik}\left(\tilde{\boldsymbol{a}}_k - \boldsymbol{b}_k^a - \boldsymbol{\eta}_k^{ad}\right)\Delta t^2\right] \end{cases} \tag{4.7}$$

Therefore, the IMU constraints between image frames can be obtained by the IMU measurements between two adjacent image frames.

https://doi.org/10.1017/S0373463320000582 Published online by Cambridge University Press

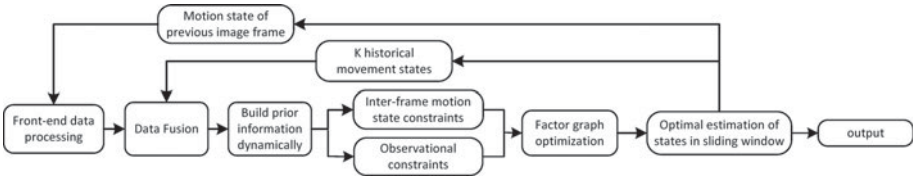


***Figure 3.*** *The processing flow of SWF.*

### *4.3. The construction of prior information in the back-end part*

Based on the front-end IMU pre-integration method, the IMU measurements are aligned with the image frames, and the processed data collected from IMU and visual sensor are sent to the back end. The back-end part limits the amount of data with the sliding window and constructs the graph with the fixed number of states in the window using factor graph optimisation method. At the same time, the prior information of the current sliding window is set dynamically to maintain the interframe constraint. The processing flow of SWF is shown in Figure 3.

Once the image frame has been acquired, the front-end data processing module outputs observation information and motion state of the current image frame and integrates it into the sliding window as the current frame information. In order to maintain a fixed number of $K$ image frames within a window, the historical image frames in the window must be processed. Then, the oldest image frame in the window is removed. Therefore, once a new image frame has been acquired, the current frame information is added to the rear end of the window, and the oldest image frame at the front of the window is removed at the same time.

If the oldest image frame in the window is directly removed and only the internal states in the window are optimised, then the inter-frame constraint will be lost. That is, there is an IMU constraint between the image frame removed and the adjacent image frame in the window. Only performing optimisation operations on the internal states in the window will easily cause the optimised states not meet the IMU constraint, which affects the overall estimate precision.

Therefore, the oldest image frame cannot be directly removed. The constraint between the oldest and the adjacent image frame in the window needs to be considered. Once new image frame has been acquired, the image frame to be removed is determined and the motion state corresponding to the image frame removed is selected as the priori information. Since the state corresponding to the removed image frame has undergone multiple optimisation estimates, it can be considered that the state corresponding to the removed image frame can be fixed without being optimised again. Select it as the prior information of current sliding window and form the IMU constraint with the state corresponding to the adjacent image frame. In the subsequent optimisation, it can be used to constrain the states to be optimised, remove the error measurement information and fix the problem of states mutation caused by the input of wrong states, so that ensure the accuracy of the estimates.

The triangulation method (Richard and Zisserman, 2000) is used to estimate the spatial positions of $M$ landmarks observed by the $K$ image frames in the window to form interframe observation constraints. Combined with the estimations of motion states, interframe motion constraints and observation constraints are established. The corresponding logical relationship is constructed using factor graphs, and the numerical iterative method is used to approximate the optimal solutions of $K$ motion states in the window. Then, the motion state corresponding to the oldest image frame of current window is selected and used as the output of current sliding window.

### *4.4. The optimisation of the states in the window*

Based on the data processing of the current sliding window, $K$ motion states, $M$ spatial positions of landmarks and the prior information of current sliding window are obtained. The graph is constructed with the data obtained. Then the optimisation of graph is equivalent to the least squares problem inside

https://doi.org/10.1017/S0373463320000582 Published online by Cambridge University Press

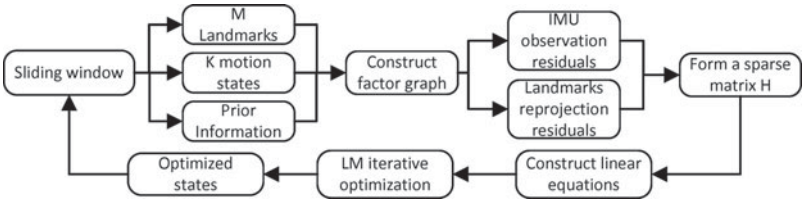


***Figure 4.*** *The processing flow of factor graph optimisation.*

the window. For the established graph, use the LM method to obtain the optimal solution of each factor node through a series of linear approximations. The processing flow of factor graph optimisation is shown in Figure 4.

The state $\boldsymbol{x}$ is constituted with $K$ motion states $\boldsymbol{x}_n$ in current sliding window, $M$ landmarks $\boldsymbol{l}_n$ observed by the $K$ motion states, as well as the priori information $\boldsymbol{x}_{\text{pri}}$, which can be written as

$$\boldsymbol{x} = \{\boldsymbol{x}_{\text{pri}}, \boldsymbol{x}_1, \ldots, \boldsymbol{x}_K, \boldsymbol{l}_1, \ldots, \boldsymbol{l}_M\} \tag{4.8}$$

where the state $\boldsymbol{x}_n$ includes the position and rotation of the carrier, $\boldsymbol{l}_m = \begin{bmatrix} x_m & y_m & z_m \end{bmatrix}$ and ${}^{\text{c}}\boldsymbol{l}_m = \begin{bmatrix} {}^{\text{c}}x_m & {}^{\text{c}}y_m & {}^{\text{c}}z_m \end{bmatrix}$ refer to the spatial positions of the $m^{\text{th}}$ landmark in the world coordinate and camera coordinate system, respectively. Showing non-zero blocks only, the sparse matrix $\boldsymbol{H}$ is constructed as

$$\boldsymbol{H} = \begin{bmatrix} -\boldsymbol{H}_{\boldsymbol{x},1} & 1 & & & & -\boldsymbol{G}_{\boldsymbol{l},1} \\ & -\boldsymbol{G}_{\boldsymbol{x},1} & & & & \\ & & \cdots & \cdots & \cdots & \\ & & -\boldsymbol{H}_{\boldsymbol{x},K} & 1 & & \\ & & & -\boldsymbol{G}_{\boldsymbol{x},K} & -\boldsymbol{G}_{\boldsymbol{l},K} & \end{bmatrix} \tag{4.9}$$

where $\boldsymbol{H}_{\boldsymbol{x},n}$ is a $6 \times 6$ Jacobian matrix

$$\boldsymbol{H}_{\boldsymbol{x},n} = \begin{bmatrix} \dfrac{\partial \boldsymbol{e}^n_{\text{IMU}}}{\partial \boldsymbol{p}_n} & \dfrac{\partial \boldsymbol{e}^n_{\text{IMU}}}{\partial \boldsymbol{R}_n} \end{bmatrix} = \begin{bmatrix} \dfrac{\partial {}^{\text{P}}\boldsymbol{e}^n_{\text{IMU}}}{\partial \boldsymbol{p}_n} & \dfrac{\partial {}^{\text{P}}\boldsymbol{e}^n_{\text{IMU}}}{\partial \boldsymbol{R}_n} \\ \dfrac{\partial {}^{\text{R}}\boldsymbol{e}^n_{\text{IMU}}}{\partial \boldsymbol{p}_n} & \dfrac{\partial {}^{\text{R}}\boldsymbol{e}^n_{\text{IMU}}}{\partial \boldsymbol{R}_n} \end{bmatrix} \tag{4.10}$$

where IMU error $\boldsymbol{e}^n_{\text{IMU}}$ = difference between the pre-integration term and the state solved by the IMU measurements at epoch $n$, including position error ${}^{\text{P}}\boldsymbol{e}^n_{\text{IMU}}$ and rotation error ${}^{\text{R}}\boldsymbol{e}^n_{\text{IMU}}$; $\boldsymbol{R}_n$ = rotation matrix at epoch $n$; $\boldsymbol{p}_n$ = the position at epoch $n$; and $\boldsymbol{G}_{\boldsymbol{x},n}$ = a $2m \times 6$ matrix, where $m$ is determined by the number of landmarks observed in the current observation frame, that is

$$\boldsymbol{G}_{\boldsymbol{x},n} = [\boldsymbol{G}^1_{\boldsymbol{x},n}, \ldots, \boldsymbol{G}^m_{\boldsymbol{x},n}]^{\text{T}} \tag{4.11}$$

where $\boldsymbol{G}^m_{\boldsymbol{x},n}$ is a $2 \times 6$ Jacobian matrix

$$\boldsymbol{G}^m_{\boldsymbol{x},n} = \begin{bmatrix} \dfrac{\partial \boldsymbol{e}^m_{\text{VIS}}}{\partial \boldsymbol{p}_n} & \dfrac{\partial \boldsymbol{e}^m_{\text{VIS}}}{\partial \boldsymbol{R}_n} \end{bmatrix} = \begin{bmatrix} \dfrac{\partial \boldsymbol{e}^m_{\text{VIS}}}{\partial {}^{\text{c}}\boldsymbol{l}_m} \dfrac{\partial {}^{\text{c}}\boldsymbol{l}_m}{\partial \boldsymbol{p}_n} & \dfrac{\partial \boldsymbol{e}^m_{\text{VIS}}}{\partial {}^{\text{c}}\boldsymbol{l}_m} \dfrac{\partial {}^{\text{c}}\boldsymbol{l}_m}{\partial \boldsymbol{R}_n} \end{bmatrix} \tag{4.12}$$

https://doi.org/10.1017/S0373463320000582 Published online by Cambridge University Press

where $\boldsymbol{e}_{\mathrm{VIS}}^{m} = \left[ {}^{\mathrm{u}}\boldsymbol{e}_{\mathrm{VIS}}^{m} \ {}^{\mathrm{v}}\boldsymbol{e}_{\mathrm{VIS}}^{m} \right]$ = observation error of the $m^{\mathrm{th}}$ landmark projected in the pixel coordinate system at epoch $n$.

$$\boldsymbol{g} = \frac{\partial \boldsymbol{e}_{\mathrm{VIS}}^{m}}{\partial {}^{\mathrm{c}}\boldsymbol{l}_{m}} = \begin{bmatrix} \dfrac{\partial {}^{\mathrm{u}}\boldsymbol{e}_{\mathrm{VIS}}^{m}}{\partial {}^{\mathrm{c}}x_{m}} & \dfrac{\partial {}^{\mathrm{u}}\boldsymbol{e}_{\mathrm{VIS}}^{m}}{\partial {}^{\mathrm{c}}y_{m}} & \dfrac{\partial {}^{\mathrm{u}}\boldsymbol{e}_{\mathrm{VIS}}^{m}}{\partial {}^{\mathrm{c}}z_{m}} \\ \dfrac{\partial {}^{\mathrm{v}}\boldsymbol{e}_{\mathrm{VIS}}^{m}}{\partial {}^{\mathrm{c}}x_{m}} & \dfrac{\partial {}^{\mathrm{v}}\boldsymbol{e}_{\mathrm{VIS}}^{m}}{\partial {}^{\mathrm{c}}y_{m}} & \dfrac{\partial {}^{\mathrm{v}}\boldsymbol{e}_{\mathrm{VIS}}^{m}}{\partial {}^{\mathrm{c}}z_{m}} \end{bmatrix} \tag{4.13}$$

$\boldsymbol{G}_{\boldsymbol{l},n}$ is a $2m \times 6m$ matrix, where $m$ is determined by the number of landmarks observed in the current observation frame. $\boldsymbol{G}_{\boldsymbol{l},n}$ is expressed as

$$\boldsymbol{G}_{\boldsymbol{l},n} = \begin{bmatrix} \boldsymbol{G}_{\boldsymbol{l},n}^{1} & & & \\ & \boldsymbol{G}_{\boldsymbol{l},n}^{2} & & \\ & & \cdots & \\ & & & \boldsymbol{G}_{\boldsymbol{l},n}^{m} \end{bmatrix} \tag{4.14}$$

where $\boldsymbol{G}_{\boldsymbol{l},n}^{m}$ is a $2 \times 3$ Jacobian matrix

$$\boldsymbol{G}_{\boldsymbol{l},n}^{m} = \frac{\partial \boldsymbol{e}_{\mathrm{VIS}}^{m}}{\partial \boldsymbol{l}_{m}} = \begin{bmatrix} \dfrac{\partial \boldsymbol{e}_{\mathrm{VIS}}^{m}}{\partial {}^{\mathrm{c}}\boldsymbol{l}_{m}} \dfrac{\partial {}^{\mathrm{c}}\boldsymbol{l}_{m}}{\partial \boldsymbol{l}_{m}} & \dfrac{\partial \boldsymbol{e}_{\mathrm{VIS}}^{m}}{\partial {}^{\mathrm{c}}\boldsymbol{l}_{m}} \dfrac{\partial {}^{\mathrm{c}}\boldsymbol{l}_{m}}{\partial \boldsymbol{l}_{m}} \end{bmatrix} \tag{4.15}$$

The $K$ IMU observation errors $\boldsymbol{e}_{\mathrm{IMU}}^{n}$ and $M$ landmarks observation errors $\boldsymbol{e}_{\mathrm{VIS}}^{m}$ are stacked to form the internal error vector $\boldsymbol{e}$ in current sliding window. $\boldsymbol{\Sigma}$ = total covariance, the structure is

$$\boldsymbol{e} = \begin{bmatrix} \boldsymbol{e}_{\mathrm{IMU}}^{1} \\ \cdots \\ \boldsymbol{e}_{\mathrm{IMU}}^{K} \\ \boldsymbol{e}_{\mathrm{VIS}}^{1} \\ \cdots \\ \boldsymbol{e}_{\mathrm{VIS}}^{M} \end{bmatrix}, \boldsymbol{\Sigma}^{-1} = \begin{bmatrix} \boldsymbol{\Sigma}_{0}^{-1} & & & & & \\ & \cdots & & & & \\ & & \boldsymbol{\Sigma}_{K}^{-1} & & & \\ & & & \boldsymbol{\Sigma}_{K+1}^{-1} & & \\ & & & & \cdots & \\ & & & & & \boldsymbol{\Sigma}_{K+M}^{-1} \end{bmatrix} \tag{4.16}$$

Form linear equations corresponding to all states in current sliding window for linear optimisation

$$(\boldsymbol{H}^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}\boldsymbol{H})\delta\boldsymbol{x} = -\boldsymbol{H}\boldsymbol{\Sigma}^{-1}\boldsymbol{e} \tag{4.17}$$

According to the construction of factor graphs, motion state estimations in the current sliding window are determined jointly by the motion states, landmark positions and prior information. Since the state related to the prior information is fixed, the algorithm does not need to solve the increment related to the fixed value during optimisation process. Then the dimension of matrix $\boldsymbol{H}$ can be reduced and the influence of prior information can be kept, so that the computational complexity is reduced, and the incremental solution is not affected.

The overall state increment is obtained by solving the linear equation, and the node state is corrected multiple times using the overall state increment. Combined with the prior information of the current sliding window, the optimal estimations of states in the window can be obtained and fed back to the window for the next optimisation operation.

## 5. Experimental results and analysis

### *5.1. Experimental environment and design*

To validate the new method, the comparative experiments are designed between the optimal filtering methods and factor graph optimisation method. Based on the EKF, InEKF and the proposed DSWFG method, the karlsruhe institute of technology and toyota institute of technology (KITTI) dataset is

https://doi.org/10.1017/S0373463320000582 Published online by Cambridge University Press

***Table 1.*** *Experimental parameters.*

| Simulation parameters | Value | Unit |
|---|---|---|
| Gyroscope noise | 0·04 | $\mathrm{rad}/\left(\mathrm{h}\sqrt{\mathrm{Hz}}\right)$ |
| Gyroscope random walk | 1·0e-06 | $\mathrm{rad}/\left(\mathrm{s}^2\sqrt{\mathrm{Hz}}\right)$ |
| Accelerometer noise | 0·04 | $\mathrm{m}/\left(\mathrm{s}^2\sqrt{\mathrm{Hz}}\right)$ |
| Accelerometer random walk | 1·0e-06 | $\mathrm{m}/\left(\mathrm{s}^3\sqrt{\mathrm{Hz}}\right)$ |
| IMU initialise attitude variance | 1·0e-06 | $\mathrm{rad}^2$ |
| IMU initialise position variance | 1·0e-04 | $\mathrm{m}^2$ |
| IMU initialise velocity variance | 1·0e-06 | $(\mathrm{m/s})^2$ |

selected as the experimental environment. The KITTI dataset is a combination of image frames, trajectory data and IMU data collected by the autonomous driving platform driving in different complex scenarios, such as cities, villages and highways (Geiger et al., 2012). The experiments mainly use the grayscale images provided by the dataset fragments, the IMU measurements provided by the OXTS RT 3003 positioning system, and the real position and attitude data to test the effect of the algorithm. The synchronised IMU output rate and image output rate are both 10Hz.

In this paper, the motion trajectory estimation, pose error statistics, and single-cycle time consuming are analysed to verify the proposed DSWFG algorithms. The setting of the sliding window range and the maximum number of optimisation iterations greatly affect the accuracy and real-time performance of the DSWFG algorithm. In order to ensure that the experiment results in different scenes are not affected by the change of parameters, the parameters of the sliding window range and the maximum number of iterations are fixed. Then, the range is set to 10, and the maximum number is set to 6. And the simulation is processed in the Matlab R2016 on a PC with Intel Core i5-9400 CPU at 2·90 GHz, 8-GB RAM equipped with Windows 10. The main noise parameters and variances are shown in Table 1.

### *5.2. Multi-scenario comparison experiments*

#### *5.2.1. The residential area driving scene*

The dataset sequence named 2011_09_26_drive_0036 is used. In order to establish a table for further quantitative analysis, the dataset segment name is abbreviated as 26–36. The dataset contains 803 image frames and a total of 715 meters of residential area driving. This scene contains the driving process of two right-angle turns. Figure 5 intuitively describes the results of the EKF, InEKF and DSWFG algorithms on this dataset segment. The lines in Figure 5(a) represent the real motion trajectory Vs. the estimated trajectory of each algorithm. In Figures 5(b) and 5(c), each algorithm is evaluated by three-directional position errors and attitude angle errors.

Before the second right-angle bend, it can be seen that the motion trajectories estimated by the three algorithms are basically consistent with the real trajectory, and the main error lies in the turning of the real trajectory. At the second right-angle turn, which is at frame 338, the driving platform encounters and follows a truck. Because the distance between the driving platform and the truck is relatively close, a huge moving object affects the constraints of landmarks. Then, it makes the position drift using the EKF and InEKF. It can also be seen that the InEKF and EKF have a certain degree of divergence in attitude and position estimation error from the frame 338. In comparison, the InEKF is superior to the EKF. And from an overall perspective, the DSWFG algorithm achieves optimal stability and superior estimation accuracy in this scenario.

https://doi.org/10.1017/S0373463320000582 Published online by Cambridge University Press

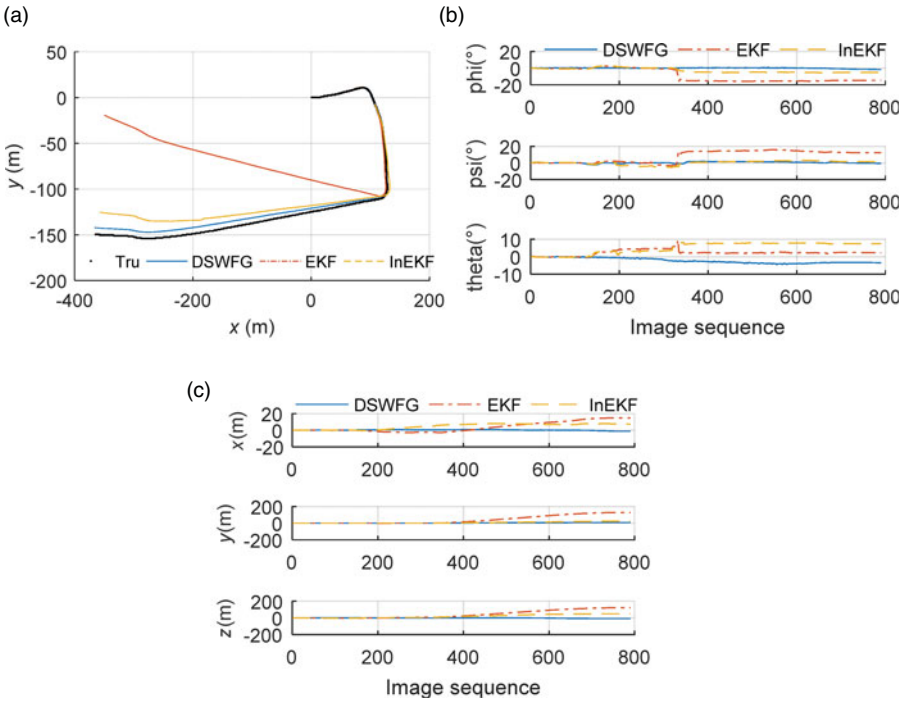


***Figure 5.*** *Results of residential area scene using the 26–36 dataset. (a) The real trajectory vs. calculated trajectories. (b) The attitude error. (c) The position error.*

*5.2.2. The city driving scene*

The dataset sequence named 2011_09_30_drive_0018 is used, and the dataset segment name is abbreviated as 30–18. The first 1200 image frames and a total of 872 meters of city driving are used for simulation. This scene contains the driving process of four right-angle turns. During this traversal, Figure 6 intuitively describes the results of the EKF, InEKF and DSWFG algorithms on this dataset segment.

Before the third right-angle bend, it can be seen that the motion trajectories estimated by the three algorithms are basically consistent with the real trajectory. At the 887th frame and the 1134th frame around the fourth right-angle turn, there are moving objects, which last about 30 frames and 20 frames, respectively. The common effect of moving objects and right-angle bending causes the divergence of the trajectory estimated by the InEKF and leads to worse estimation accuracy. Due to the accumulation of errors, the estimation accuracy of EKF is much lower than the DSWFG algorithm. At the same time, it can also be seen from Figures 6(b) and 6(c) that the position estimation error of the InEKF starts to diverge from the 1134th frame, while the EKF algorithm and the DSWFG algorithm can still maintain good stability. The overall comparison shows that the DSWFG algorithm achieves the best stability and superior estimation accuracy in this scenario.

*5.2.3. The highway driving scene*

The dataset sequence named 2011_10_03_drive_0042 is used, and the dataset segment name is abbreviated as 03–42. The 1170 image frames and a total of 2566 meters of highway driving area are used. This scene records the driving process of getting on and off the highway, which includes a starting curve and a final 360-degree circular curve. In this traversal process, Figure 7 describes the results of the EKF, InEKF and DSWFG algorithms. Given that the number of vehicles on the highway are few and other vehicles are relatively fast in this dataset fragment, the impact of moving objects is relatively small.

https://doi.org/10.1017/S0373463320000582 Published online by Cambridge University Press

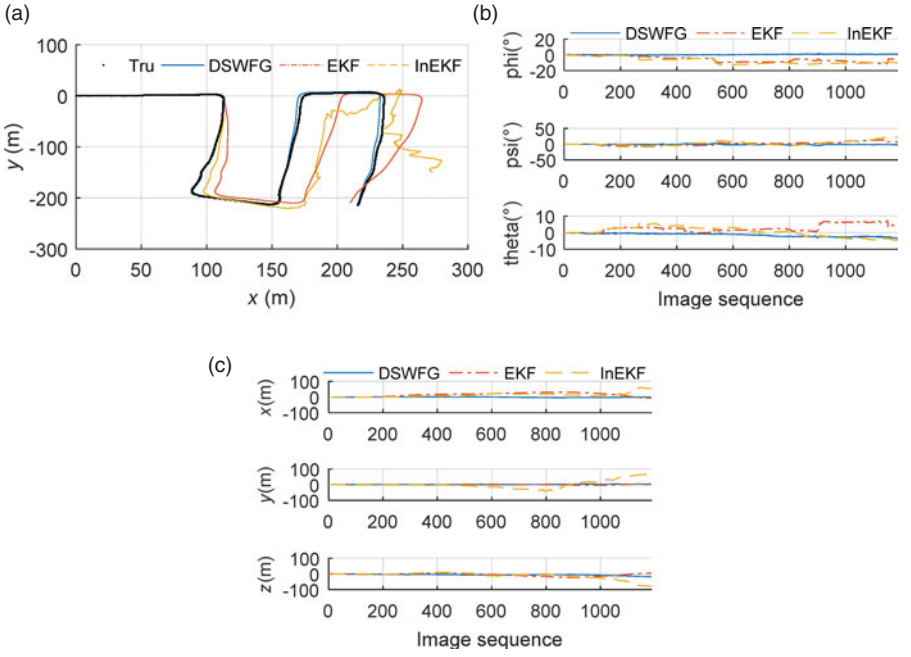


***Figure 6.*** *Results of city driving scene using the 30–18 dataset. (a) The real trajectory vs. calculated trajectories. (b) The attitude error. (c) The position error.*

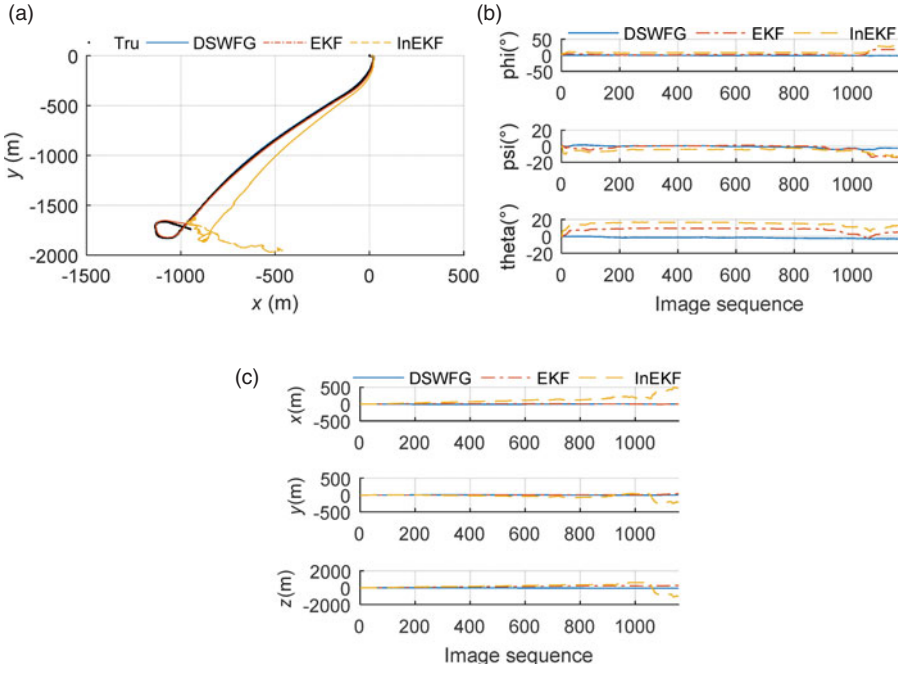


***Figure 7.*** *Results of highway driving scene using the 03–42 dataset. (a) The real trajectory vs. calculated trajectories. (b) The attitude error. (c) The position error.*

https://doi.org/10.1017/S0373463320000582 Published online by Cambridge University Press

***Table 2.*** *Results of EKF, InEKF and DSWFG.*

| Sequence | Method | RMSE | | | | | | Average |
|---|---|---|---|---|---|---|---|---|
| | | x (m) | y (m) | z (m) | Pitch (°) | Roll (°) | Yaw (°) | time (s) |
| 26-19 | EKF | **0·356** | 4·288 | 6·285 | 2·158 | 1·710 | 6·810 | 0·095 |
| | InEKF | 0·821 | 2·938 | 9·091 | 1·449 | 1·254 | 1·518 | **0·037** |
| | DSWFG | 1·198 | **2·389** | **2·785** | **1·364** | **0·151** | **0·739** | 0·592 |
| 26-36 | EKF | 7·150 | 64·603 | 64·007 | 11·526 | 2·755 | 10·623 | 0·099 |
| | InEKF | 5·802 | 11·944 | 28·520 | 3·801 | 6·053 | 2·315 | **0·039** |
| | DSWFG | **0·525** | **4·413** | **3·751** | **0·553** | **2·626** | **0·981** | 0·549 |
| 30-18 | EKF | 19·132 | 2·152 | 10·081 | 6·445 | 3·477 | 5·878 | 0·120 |
| | InEKF | 18·314 | 24·761 | 23·087 | 8·502 | 2·894 | 7·772 | **0·052** |
| | DSWFG | **1·856** | **1·864** | **6·894** | **0·615** | **1·663** | **1·464** | 0·373 |
| 30-20 | EKF | **1·894** | 22·954 | 11·088 | 3·333 | 5·745 | 6·268 | 0·109 |
| | InEKF | 15·777 | 56·610 | 11·757 | 2·410 | 7·193 | 14·599 | **0·054** |
| | DSWFG | 2·720 | **5·315** | **6·840** | **0·882** | **1·374** | **1·486** | 0·406 |
| 30-33 | EKF | 154·187 | 112·883 | 84·057 | 11·862 | 6·627 | 30·531 | 0·135 |
| | InEKF | 208·391 | 178·597 | 157·354 | 7·328 | 10·683 | 41·192 | **0·070** |
| | DSWFG | **4·982** | **6·337** | **21·047** | **0·496** | **0·324** | **1·168** | 0·250 |
| 30-34 | EKF | 40·222 | 80·776 | 111·373 | 38·377 | 43·699 | 9·970 | 0·119 |
| | InEKF | 30·303 | 29·229 | 144·733 | 4·982 | 3·732 | 7·031 | **0·054** |
| | DSWFG | **9·303** | **2·544** | **7·733** | **0·464** | **0·554** | **0·100** | 0·313 |
| 03-42 | EKF | 8·448 | 9·217 | 167·231 | 5·681 | 8·237 | 4·355 | 0·097 |
| | InEKF | 161·326 | 70·874 | 362·543 | 11·288 | 15·008 | 5·699 | **0·045** |
| | DSWFG | **5·163** | **3·290** | **36·903** | **0·816** | **1·809** | **1·696** | 0·485 |

It can be seen that the movement trajectories solved by three verified algorithms are basically consistent with the real trajectory in Figure 7(a). The motion trajectories solved by the DSWFG algorithm and EKF algorithm almost match the real trajectory, especially in the middle of the real trajectory. Due to the influence of dynamic objects at the beginning, the trajectory obtained by the InEKF is deviated, which in turn affects the accuracy of the subsequent trajectory.

*5.2.4. Statistical analysis*

In addition to the typical three sets of experimental results above, more experiments were carried out for quantitative analysis. Table 2 shows the error statistical results of seven driving scenes, including rural roads, circular driving, complex loops and highways. The bold part indicates that the result is relatively optimal. Based on the comparison results of the three algorithms, it can be seen that the DSWFG algorithm can still maintain a good solution result in various sports scenarios. In order to compare the results of the three algorithms accurately, the root-mean-square error (RMSE) in three directions and the computing time are taken as the evaluation index. The following conclusions can be drawn from Table 2:

(a) From the results of sequences 26–36, 30–18, 30–33, 30–34 and 03–42, it can be seen that the DSWFG algorithm has better accuracy than the InEKF and EKF algorithms in terms of position and attitude errors.
(b) From the results of sequences 26–19 and 30–20, it can be seen the EKF are slightly better than the DSWFG algorithm in some ways. However, considering the overall estimation accuracy of the position and attitude, the DSWFG algorithm has better accuracy than do the InEKF and EKF algorithms.

https://doi.org/10.1017/S0373463320000582 Published online by Cambridge University Press

(c) Analysing the time consuming of a single cycle, an overall comparison shows that the InEKF algorithm is the fastest, while the DSWFG algorithm is relatively slow.
(d) The real-time performance of the InEKF is superior to the other two algorithms, and its accuracy is generally better than the traditional EKF method under the ideal conditions during the background environment is stationary. However, with the influence of moving objects, the stability of the InEKF is poor, and its motion trajectory is easy to diverge. The real-time performance of the DSWFG is relatively poor, but the comparison results in various motion scenes show that it can maintain the superior accuracy and optimal stability.

## 6. Conclusion

This paper presents a new visual–inertial integrated navigation algorithm. Considering the optimisation processing of factor graphs, the front end of the algorithm uses the IMU pre-integration to align the IMU with the image frame. Because of the effect of data accumulation and prior information on the optimisation performance of factor graphs, the DSWFG algorithm is proposed. The sliding window constraint is added to the factor graphs to limit the operation scale, the prior information of the sliding window is dynamically set to maintain the interframe constraints, and the error measurements are corrected to ensure the accuracy of the optimised quantities in the sliding window. A series of dynamic experiments of the residential area, the city driving and the highway driving scenes are carried out to demonstrate the new approach. The traditional EKF, the improved InEKF and the new DSWFG algorithms are carried out for comparison through the aspects of motion trajectory, attitude, and positioning error and time consuming. The results show that DSWFG algorithm can maintain the superior solution accuracy and optimal stability in various motion scenes. This research provides a reference for the further study of visual–inertial navigation systems.

**Acknowledgements.** Research for this paper was supported by the Fundamental Research Project for the Central Universities (NS2019047). The authors also gratefully acknowledge the helpful comments and suggestions of the reviewers, which have improved the presentation.

## References

**Barrau, A.** (2015). *Non-linear state error based extended Kalman filters with applications to navigation* (PhD thesis). Mines Paristech.

**Dellaert, F. and Kaess, M.** (2017). Factor graphs for robot perception. *Foundations and Trends in Robotics*, **6**(1–2), 1–139.

**Dong-Si, T. and Mourikis, A. I.** (2011). Motion Tracking with Fixed-lag Smoothing: Algorithm and Consistency Analysis. *2011 IEEE International Conference on Robotics and Automation*. Shanghai, 5655–5662.

**Eckenhoff, K., Geneva, P. and Huang, G.** (2017). Direct Visual-Inertial Navigation With Analytical Preintegration. *2017 IEEE International Conference on Robotics and Automation (ICRA)*, Singapore, Singapore, 1429–1435.

**Forster, C., Carlone, L., Dellaert, F. and Scaramuzza, D.** (2017). On-manifold preintegration for real-time visual–inertial odometry. *IEEE Transactions on Robotics*, **33**(1), 1–21.

**Geiger, A., Lenz, P. and Urtasun, R.** (2012). Are We Ready for Autonomous Driving? The KITTI Vision Benchmark Suite. *2012 IEEE Conference on Computer Vision and Paattern Recognition(CVPR)*. RI, USA: IEEE, 3354–3361.

**Geiger, A., Lenz, P., Stiller, C. and Urtasun, R.** (2013). Vision meets Robotics: the KITTI Dataset. *The Internaltional Journal of Robotics Research(IJRR*, **32**(11), 1231–1237.

**Hesch, J. A., Kottas, D. G., Bowman, S. L. and Roumeliotis, S. I.** (2014). Consistency analysis and improvement of vision-aided inertial navigation. *IEEE Transactions on Robotics*, **30**(1), 158–176.

**Hsiung, J., Hsiao, M., Westman, E., Valencia, R. and Kaess, M.** (2018). Information Sparsification in Visual-Inertial Odometry. *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Madrid, Spain, 1146–1153.

**Huang, G.** (2019). Visual-Inertial Navigation: A Concise Review. *2019 International Conference on Robotics and Automation (ICRA)*. Montreal, Canada, 9572–9582.

**Huang, G. P., Mourikis, A. I. and Roumeliotis, S. I.** (2009). A First-Estimates Jacobian EKF for Improving SLAM Consistency. *Experimental Robotics*, Berlin, 373–382.

**Jiang, J., Niu, X., Guo, R. and Liu, J.** (2019). A hybrid sliding window optimizer for tightly-coupled vision-aided inertial navigation system. *Sensors (Basel)*, **19**(15), 3418–3439.

**Johannsson, H., Kaess, M., Fallon, M. and Leonard, J. J.** (2013). Temporally Scalable Visual SLAM Using a Reduced Pose Graph. In *Proceedings of the IEEE International Conference on Robotics and Automation(ICRA)*, Karlsruhe, 54–61.

**Kretzschmar, H., Stachniss, C. and Grisett, G.** (2011). Efficient Information-Theoretic Graph Pruning for Graph-Based SLAM With Laser Range Finders. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, San Francisco, 865–871.

**Li, M. and Mourikis, A. I.** (2013). High-precision, consistent EKF-based visual-inertial odometry[J]. *The International Journal of Robotics Research*, **32**(6), 690–711.

**Mourikis, A. I. and Roumeliotis, S. I.** (2007). A Multi-State Constraint Kalman Filter for Vision-Aided Inertial Navigation. *Proceedings 2007 IEEE International Conference on Robotics and Automation*, Italy, 3565–3572.

**Richard, H. and Zisserman, A.** (2000). *Multiple View Geometry in Computer Vision[M]*. New York, UK: Cambridge University Press, 217–220.

**Sibley, G., Sukhatme, G. and Matthies, L.** (2006). The iterated sigma point Kalman filter with applications to long range stereo. *In Robotics:Science and Systems*, **8**, 263–270.

**Sibley, G., Matthies, L. and Sukhatme, G. S.** (2010). Sliding window filter with application to planetary landing. *Journal of Field Robotics*, **27**(5), 1556–4959.

**Wilbers, D., Rumberg, L. and Stachniss, C.** (2019). Approximating Marginalization with Sparse Global Priors for Sliding Window SLAM-Graphs. *2019 Third IEEE International Conference on Robotic Computing (IRC)*, Italy, 25–31.

**Wu, K., Zhang, T., Su, D., Huang, S. and Dissanayake, G.** (2017). An invariant-EKF VINS algorithm for improving consistency. *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Vancouver, BC, 1578–1585.

**Zhuang, Y., Wang, Q., Li, Y., Gao, Z., Zhou, B., Qi, L., Yang, J., Chen, R. and El-Sheimy, N.** (2019). The integration of photodiode and camera for visible light positioning by using fixed-lag ensemble Kalman smoother. *Remote Sensing*, **11**(11), 1387.

**Cite this article:** Liu H, Wang J, Feng J, Wang X (2021). A new visual/inertial integrated navigation algorithm based on sliding-window factor graph optimisation. *The Journal of Navigation* **74**: 2, 343–359. https://doi.org/10.1017/S0373463320000582